

# Multiscale Abstraction, Planning and Control using Diffusion Wavelets for Stochastic Optimal Control Problems

Jung-Su Ha and Han-Lim Choi

**Abstract**—This work presents a multiscale framework to solve a class of stochastic optimal control problems in the context of robot motion planning & control in a complex environment. In order to handle complications due to large decision space and complex environmental geometry, two key concepts are adopted: (a) diffusion wavelet representation of Markov chain for hierarchical abstraction of the state space; and (b) desirability function-based representation of a Markov decision process (MDP) for efficient calculation of optimal policy. In the proposed framework, a global plan that compressively takes into account long time/length-scale state transition is first obtained by approximately solving an MDP whose desirability function is represented by coarse scale bases in the hierarchical abstraction; then, a detailed local plan is computed by solving an MDP that considers wavelet bases associated with a focused region of the state space, guided by the global plan. The resulting multiscale plan is utilized to finally compute a continuous-time optimal control policy within a receding horizon implementation. Two numerical examples are presented to demonstrate the applicability and validity of the proposed approach.

## I. INTRODUCTION

In this work, we address a continuous time/continuous state stochastic optimal control (SOC) problem for robots operated in a complex environment over a long time horizon. SOC is the problem of computing the optimal policy for a system driven by uncertain disturbances to maximize a certain performance index. A standard and general way to obtain the optimal control solution is the dynamic programming approach, which computes the optimal cost-to-go function (also called the value function) for all possible state and time, and then reconstruct a control policy from the value function. In a continuous-time/continuous-state problem, a nonlinear partial differential equation called the Hamilton-Jacobi-Bellman (HJB) equation needs to be solved, which is computationally intractable in most robotic control applications.

There is a class of SOC, called the *linearly-solvable optimal control (LSOC)*, for which the HJB equation can naturally be linearized and thus efficient solution methods exist [1,2]. In LSOC, the exponentiated value function, termed the *desirability function*, is used as the principal eigenfunction of the linearized differential operator. Due to the linearity structure in LSOC, several effective solution schemes have been addressed: (a) to approximate the eigenstructure with some generic function approximation techniques such as Gaussian radial basis function (RBF) [3], or (b) to obtain the distribution of desirability functions by sampling many

continuous trajectories and evaluating them, which is called the path integral control [1]. However, if the LSOC problem of interest is associated with a very long time horizon and/or complex/high-dimensional geometric domains (e.g., induced by obstacles), the aforementioned techniques cannot readily be implemented since obtaining an appropriate basis set for function approximation becomes non-trivial and simulating trajectories (and checking potential collisions) becomes computationally expensive [3]–[5]. If the original continuous SOC problem is discretized, it can be represented as a Markov decision process (MDP) and solved using well-developed iterative methods such as policy iteration and value iteration. However, this discretization approach is inherently subject to curse of dimensionality, which imposing a significant limit in handling high-dimensional problems. There is a discrete-time equivalent of LSOC, which is called linearly-solvable MDP (LMDP), but solution methodologies for this class of problem still exhibit limitation in handling high-dimensional (and long-horizon) problems.

To address a large-scale decision problems effectively, it is conceivable that the hallmark of human intelligence could provide some insights - in particular, this work notes *multiscale* and *hierarchical* structure of human decision making. Suppose that someone currently writing a paper at his/her office desk wants to get out of the building; the office is located in the third floor of the building and there is one set of staircases and an elevator. Then, what would this person's control policy look like? This person would not try to figure out what he/she should do for all possible situations he/she could face like the standard value function-based approach; instead, he/she would figure out which building gate he/she would use, whether to take the elevator or the stairs, which door he would exit from the room (if there are more than one), etc. A detailed plans such as "which particular start he/she should put their left foot," would be determined later in the process of executing a piece of overall plan, for example, "go downstairs using the staircase." It should be noted that this human-like decision takes advantage of the underlying (multiscale) hierarchical structure of state space; in the above example, detailed aspects such a particular certain sequence of stairs to put on is abstracted by just a single notion of staircase.

There have been various studies to obtain the hierarchical structure of spaces. One such approach that has been extensively studied is multi-resolution analysis (MRA) that is based on the wavelet theory [6]. MRA is to find a sequence of basis sets that spans a certain nested subspace; for example, if applied to a function approximation problem, the basis func-

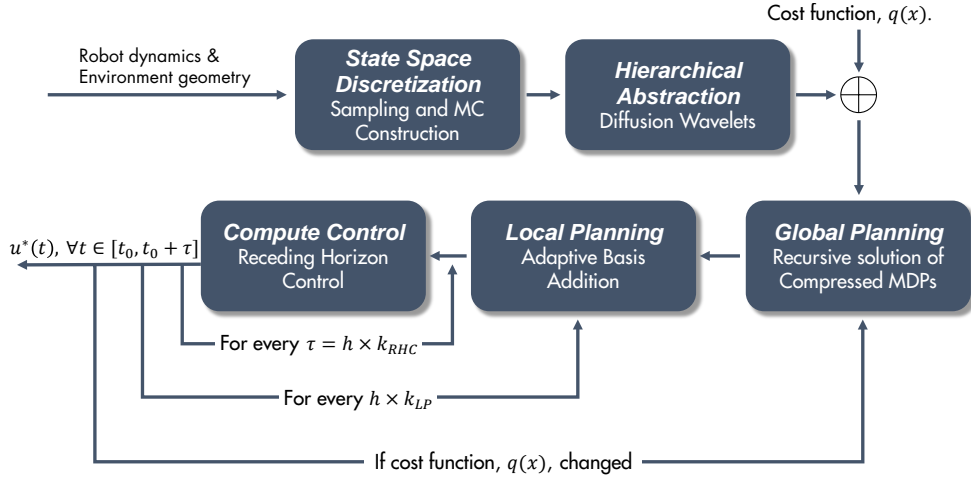


Fig. 1. Proposed framework

tions in a coarse length scale will only reconstruct the rough trend of the target function. Although this MRA method provides a systematic scheme for hierarchical abstraction, construction of the wavelet bases over a high-dimensional complex geometric domain is not straightforward. The notion of diffusion wavelet [7] is shown to provide more general wavelet bases construction procedure; this type of abstraction over a Markov chain has been utilized for expedite policy evaluation for MDP [8]. Also in representation policy iteration [9], which simultaneously learn the control and system representation during the task executed, diffusion wavelets are used to construct basis set.

This work addresses a large-scale, long time-horizon LSOC problem, taking advantage of abstraction scheme using the diffusion wavelet bases to approximate the desirability functions that naturally leads to solution of HJB equation. The key contribution of this paper is to present a systematic framework for solving LSOC, as depicted in Fig. 1; this is, to the authors' best knowledge, the first work that takes advantage of the multiscale structure of the basis set to solve an LSOC problem. The framework consists of five phases: (i) In the discretization phase, Markov chain associated to the robot dynamics is constructed by sampling a set finite set of states in state-space. (ii) In the abstraction phase, the hierarchical bases structure is obtained using the diffusion wavelet method. (iii) In global planning phase, an MDP constructed only using the coarse wavelet bases (or on "abstract-state") is solved; this MDP is much more tractable to handle than using the original bases set. Our work fundamentally differs from [8,9] in the sense that the abstraction is obtained by discretizing the given stochastic system dynamics and multiscale structure of the basis sets is utilized to solve compressed problems recursively. (iv) In the local planning phase, focused regions where the robot will most likely visit for near future are sought for and detailed policy associated with these focused regions are computed. A certain search procedure is also applied on on the coarse bases using information of optimal transition which is naturally provided by the (approximate) solution of LMDP. (vi) In the control phase, a continuous control

sequence is computed and applied to the robot in a receding horizon fashion. Rest of the paper is primarily focused on elaborating the details of this framework, followed by numerical examples for validation of the method.

## II. CONTINUOUS-TIME STOCHASTIC OPTIMAL CONTROL PROBLEM AND TIME DISCRETIZATION

Let  $\mathbf{x} \in \chi$  and  $\mathbf{u} \in U$  be a state and control vector, respectively, where a state space,  $\chi$ , and control input space,  $U$ , are a subset of  $\mathbb{R}^{d_x}$  and  $\mathbb{R}^{d_u}$ , respectively. Suppose  $\mathbf{w}$  is an  $d_u$ -dimensional Brownian motion process. Consider the stochastic dynamics of which deterministic drift term is affine in control input:

$$d\mathbf{x} = \mathbf{f}(\mathbf{x})dt + G(\mathbf{x})(\mathbf{u}dt + \sigma d\mathbf{w}) \quad (1)$$

where  $\mathbf{f} : \chi \rightarrow \mathbb{R}^{d_x}$  is the passive dynamics and  $G : \chi \rightarrow \mathbb{R}^{d_x \times d_u}$  is control transition matrix function. Let a function  $q : \chi \rightarrow \mathbb{R}$  be an instantaneous state cost rate. Then, the cost functional which we want to minimize is defined as:

$$J(\mathbf{x}) = \lim_{t_f \rightarrow \infty} \frac{1}{t_f} E \left[ \int_0^{t_f} q(\mathbf{x}(t)) + \frac{1}{2\sigma^2} \mathbf{u}(t)' \mathbf{u}(t) dt \right]. \quad (2)$$

The prime sign,  $(\cdot)'$ , throughout the paper denotes the transpose of a matrix. The problem with the cost function (2) and dynamics (1) is called the infinite horizon average cost stochastic optimal control (SOC) problem.

The continuous-time problem is hard to solve except some special cases. In general, time-axis and state space are discretized to make the problem tractable. Here, we introduce the time-axis discretized problem with a time step  $h$ . The transition probability of one step without any control input is defined as:

$$\mathbf{x}[k+1] \sim p(\cdot | \mathbf{x}[k]), \quad (3)$$

which is called the passive dynamics. If control input is applied, the transition probability is changed and written as:

$$\mathbf{x}[k+1] \sim \pi(\cdot | \mathbf{x}[k]). \quad (4)$$

The passive and controlled dynamics can be approximated as  $\mathcal{N}(\mathbf{y}; \mu(h), \Sigma(h))$ , where  $\mathcal{N}$  is a Gaussian distribution with a mean  $\mu(h)$  and covariance  $\Sigma(h)$ . Also, for small  $h$ ,

the Kullback-Leibler divergence between two distributions is approximated as  $D_{KL}(\pi(\cdot|\mathbf{x})||p(\cdot|\mathbf{x})) = \frac{h}{2\sigma^2} \mathbf{u}'\mathbf{u}$ . Therefore, the cost functional (2) is written in discrete time setting:

$$J(\mathbf{x}) = \lim_{K \rightarrow \infty} \frac{1}{K} E \left[ \sum_{k=0}^K h q(\mathbf{x}[k]) + D_{KL}(\pi(\cdot|\mathbf{x}[k])||p(\cdot|\mathbf{x}[k])) \right]. \quad (5)$$

It is well known that the solution of the discrete-time SOC (3)-(5) converges to the solution of continuous-time SOC (1)-(2) as  $h \rightarrow 0$  [2].

### III. STATE SPACE DISCRETIZATION AND MULTI-SCALE ABSTRACTION

#### A. State Space Discretization and Associated Markov Chain

In this subsection, the methods for state space discretization and the associated Markov chain construction will be discussed. One general way is to discretize the state space by grid: it is very intuitive and simple, but it can easily suffer from the curse of dimensionality and hard to adapt complex geometry of domain induced by obstacles. Another way to discretization is a sampling method. The sampling method has several advantages over the grid method; it is easy to control the number of discrete points (samples) and to utilize the heuristics methods by adapting sampling density; also, collision checking modules are easily incorporated into the discretization scheme, which is the reason that the sampling-based algorithms are extensively studied and used in the motion planning literature.

Suppose a set of discrete states  $X = \{\mathbf{x}_n\}$  is given. Then transition matrix for passive dynamics  $P$ , where  $P_{nm}$  means a transition probability from  $\mathbf{x}_n$  to  $\mathbf{x}_m$  should be determined such that

$$\mu(h) \approx \bar{\mathbf{y}}_n = \sum_m P_{nm} \mathbf{x}_m, \quad (6)$$

$$\Sigma(h) \approx \sum_m P_{nm} (\mathbf{x}_m - \bar{\mathbf{y}}_n)(\mathbf{x}_m - \bar{\mathbf{y}}_n)', \quad (7)$$

$$\sum_m P_{nm} = 1. \quad (8)$$

It requires solving linear equation with  $(|X|^2 + |X| + 1) \times |X|$  matrix for each state. Another approach determining  $P$  is approximation via Gaussian distribution as:

$$P_{nm} = \frac{\mathcal{N}(\mathbf{x}_m : \mu(h), \Sigma(h))}{\sum_{m'} \mathcal{N}(\mathbf{x}_{m'} : \mu(h), \Sigma(h))}. \quad (9)$$

Note that for each state, it only requires to compute  $|X|$  dimensional vector and normalizing it. In general,  $\mu(h)$  and  $\Sigma(h)$  are approximated as  $\mu(h) \approx \mathbf{x}_n + h\mathbf{f}(\mathbf{x}_n)$  and  $\Sigma(h) \approx h\sigma^2 G(\mathbf{x}_n)G(\mathbf{x}_n)'$ . However, it might cause some problems when  $\Sigma(h)$  is nonsingular matrix;  $\sigma^2 G(\mathbf{x}_n)G(\mathbf{x}_n)'$  is nonsingular whenever  $d_u < d_x$ . We can approximate  $\mu(h)$  and  $\Sigma(h)$  by integrating moment dynamics of linearized SDE

for  $t \in [0, h]$ :

$$\dot{\mu}(t) = A\mu(t) + \mathbf{c}, \quad (10)$$

$$\dot{\Sigma}(t) = A\Sigma(t) + \Sigma(t)A' + BB', \quad (11)$$

$$\mu(0) = \mathbf{x}_n, \quad \Sigma(0) = 0, \quad (12)$$

where  $A = \frac{d\mathbf{f}}{d\mathbf{x}}|_{\mathbf{x}=\mathbf{x}_n}$ ,  $B = \sigma G(\mathbf{x}_n)$  and  $\mathbf{c} = \mathbf{f}(\mathbf{x}_n) - A\mathbf{x}_n$ . Note that if a linear system  $(A, B)$  is controllable, which is the case for many systems, the solution of Lyapunov equation (11),  $\Sigma(t)$ , is nonsingular for all  $t > 0$ . The transition probability of this Markov chain converges to that of (1) as  $|X| \rightarrow \infty$  and  $h \rightarrow 0$  [10]. Also, one can utilize a local Gaussian distribution which truncates tails of distribution to make  $P$  sparse.

#### B. Multi-Scale Abstraction on Graph: Diffusion Wavelets

From now on, we consider  $T = P'$  for notion simplicity; then  $T_{nm}$  represents a transition probability from  $\mathbf{x}_m$  to  $\mathbf{x}_n$ . The Markov chain,  $T$ , obtained by discretizing a diffusion process ((1) with  $\mathbf{u} = 0$ ) is known to have some interesting properties: *local*, *smoothing* and *contractive* [7]. From any initial point,  $\delta_m$ , the agent (numerically) transitions to only a few its neighbors (i.e.,  $T\delta_m$  has a small support) and  $T^j\delta_m$  is a smooth probability distribution. Also since  $\|T\|_2 \leq 1$ , a dimension of a subspace,  $V_j$ , which is  $\epsilon$ -spanned by  $\{T^j\delta_m\}_{m \in X}^1$  monotonically decreases as  $j$  increases and  $V_0 \supseteq V_1 \supseteq \dots \supseteq V_j \supseteq \dots$ ; especially for an irreducible Markov chain,  $\dim(V_j) \rightarrow 1$  as  $j$  increases and a limit of  $V_j$  corresponds to the stationary distribution of the Markov chain.

Let  $W_j$  be an orthogonal complement of  $V_{j+1}$  into  $V_j$ , i.e.,  $V_j = V_{j+1} \oplus W_j$  and suppose the bases  $\Phi_j$  and  $\Psi_j$  span  $V_j$  and  $W_j$ , respectively. By using aforementioned properties of  $T$ , *Diffusion wavelets* constructs a hierarchical structure of a set of well-localized bases  $\Phi_j$  and  $\Psi_j$  called *scaling* and *wavelet functions*, respectively. In order to explain the procedure, we introduce some notations used in [7] with slight changes. Let  $[L]_{\Phi_j}^{\Phi_{j+1}}$  be the matrix representing the operator  $L$  w.r.t. the basis  $\Phi_j$  in the domain and  $\Phi_{j+1}$  in the range, and  $[B]_{\Phi_j}$  be a set of vectors  $B$  represented on a basis  $\Phi_j$ , where the columns of  $[B]_{\Phi_j}$  are the coordinates of the vectors  $B$  in the coordinates  $\Phi_j$ .

Diffusion wavelet tree starts being constructed with a fixed precision  $\epsilon > 0$ , the basis  $\Phi_0 = \{\delta_m\}_{m \in X}$  and the operator  $T_0 := [T]_{\Phi_0}^{\Phi_0} = T$  on the basis  $\Phi_0$ . Consider the set of functions  $\tilde{\Phi}_0 = \{T\delta_m\}_{m \in X}$ , which is a set of columns of  $[T]_{\Phi_0}^{\Phi_0}$ . Because  $T$  is local, these functions are well-localized. The algorithm gets a basis  $\Phi_1 = \{\phi_{1,m}\}_{m \in X_1}$  ( $X_1$  is defined as this index set) written on the basis  $\Phi_0$  by carefully orthogonalizing  $\tilde{\Phi}_0$  to preserve being well-localized and to span a subspace which is  $\epsilon$ -close to the  $\text{span}(\tilde{\Phi}_0)$ . Note that the elements of  $\Phi_1$  are coarser than the elements of  $\Phi_0$  since  $T$  is smoothing and  $|X_1| \leq |X|$  since  $T$  is contractive. The algorithm stores the information of the new basis in  $[\Phi_1]_{\Phi_0}$  and computes the *compressed* operator

<sup>1</sup>With a slight abuse of notation, we will use  $m \in X$  to denote  $\mathbf{x}_m \in X$ .

$T_1 = [T^2]_{\Phi_1}^{\Phi_1} = [\Phi_{j+1}]_{\Phi_j}' [T^2]_{\Phi_j}^{\Phi_j} [\Phi_{j+1}]_{\Phi_j}$  w.r.t.  $\Phi_1$  in the domain and the range. Also, the wavelets  $[\Psi_0]_{\Phi_0}$  are obtained similarly by looking at the columns of  $I_{(\Phi_0)} - [\Phi_1]_{\Phi_0} [\Phi_1]_{\Phi_0}'$ . The algorithm proceeds in the same fashion to get  $[\Phi_{j+1}]_{\Phi_j}$  and  $T_{j+1}$  for  $j = 1, \dots, J-1$ . A pseudo-code of the algorithm is shown in Algorithm 1.

---

**Algorithm 1** Pseudo-code for Diffusion Wavelet Tree

---

```

1: for  $j = 0, 1, \dots, J-1$  do
2:    $[\Phi_{j+1}]_{\Phi_j} \leftarrow \text{SPARSEQR}(T_j, \epsilon)$ 
3:    $T_{j+1} = [T^{2^{j+1}}]_{\Phi_{j+1}}^{\Phi_{j+1}} \leftarrow [\Phi_{j+1}]_{\Phi_j}' T_j^2 [\Phi_{j+1}]_{\Phi_j}$ 
4:    $[\Psi_j]_{\Phi_j} \leftarrow \text{SPARSEQR}(I_{(\Phi_j)} - [\Phi_{j+1}]_{\Phi_j} [\Phi_{j+1}]_{\Phi_j}', \epsilon)$ 
5: end for
6: return DWT  $\leftarrow \{[\Phi_{j+1}]_{\Phi_j}, [\Psi_j]_{\Phi_j}; \forall j = 0, \dots, J-1\}$ 

```

---

A set of basis functions at level  $j$  can be written in the original coordinate (or can be *unpacked*) as:

$$\begin{aligned} \Phi_j &= [\Phi_j]_{\Phi_0} \\ &= [\Phi_{j-1}]_{\Phi_0} [\Phi_j]_{\Phi_{j-1}} \\ &= [\Phi_1]_{\Phi_0} \cdots [\Phi_{j-1}]_{\Phi_{j-2}} [\Phi_j]_{\Phi_{j-1}}, \end{aligned} \quad (13)$$

which is represented as a  $|X| \times |X_j|$  matrix. Note that each column of  $[\Phi_j]_{\Phi_0}$  can be viewed as an “abstract-state” of the original Markov chain. The subspace with basis  $[\Phi_j]_{\Phi_0}$  is  $j\epsilon$ -close to the subspace spanned by  $\{T^{1+2+2^2+\dots+2^{j-1}} \delta_m = T^{2^j-1} \delta_m\}_{m \in X}$ ; that is, at the scale  $j$ , where  $(2^{j-1} - 1)$  steps of the scale 0 (original scale) are considered as one-step, there are only  $|X_j|$  meaningful combinations of states and each combination,  $[\Phi_j]_{\Phi_0}$ , represents “abstract-state”.

#### IV. MULTISCALE GLOBAL AND LOCAL PLANNING

##### A. Linearly-solvable MDP and linear Bellman equation

With a set of discrete states,  $X$ , the state-space as well as time-axis discretized version of SOC is formulated as the Markov decision process (MDP). Equations representing the problem have same form as (3)-(5). Because the cost is average over infinite horizon, the optimal cost-to-go value,

$$c := \min_{\pi} J^{\pi}(\mathbf{x}), \quad (14)$$

does not depend on the initial state, which is problematic since the optimal policy is reconstructed from “difference” of the cost-to-go between states. Consider the optimal cost-to-go function for the finite horizon MDP:

$$v_K(\mathbf{x}) := \min_{\pi} E \left[ \sum_{k=0}^K hq(\mathbf{x}) + D_{KL}(\pi(\cdot|\mathbf{x})||p(\cdot|\mathbf{x})) \right]. \quad (15)$$

Using  $v_K$  for  $K \rightarrow \infty$ , differential cost-to-go function is defined as:

$$v(\mathbf{x}) := v_K(\mathbf{x}) - Kc. \quad (16)$$

Then  $c$  and  $v$  satisfies the Bellman equation:

$$\begin{aligned} hc + v(\mathbf{x}) \\ = \min_{\pi} (hq(\mathbf{x}) + D_{KL}(\pi(\cdot|\mathbf{x})||p(\cdot|\mathbf{x})) + E_{\mathbf{x}' \sim \pi(\cdot|\mathbf{x})} [v(\mathbf{x}')]). \end{aligned} \quad (17)$$

By defining the (differential) desirability function,

$$z(\mathbf{x}) = \exp(-v(\mathbf{x})),$$

and the linear operator  $\mathcal{G}[z](\mathbf{x}) = \sum_{\mathbf{x}'} p(\mathbf{x}'|\mathbf{x}) z(\mathbf{x}')$ , we can rewrite the right side of the Bellman equation as:

$$\begin{aligned} \min_{\pi} \left( hq(\mathbf{x}) + E_{\mathbf{x}' \sim \pi(\cdot|\mathbf{x})} \left[ \log \left( \frac{\pi(\mathbf{x}'|\mathbf{x})}{p(\mathbf{x}'|\mathbf{x}) \exp(-v(\mathbf{x}'))} \right) \right] \right) = \\ \min_{\pi} \left( hq(\mathbf{x}) - \log \mathcal{G}[z](\mathbf{x}) + D_{KL} \left( \pi(\mathbf{x}'|\mathbf{x}) || \frac{p(\mathbf{x}'|\mathbf{x}) z(\mathbf{x}')}{\mathcal{G}[z](\mathbf{x})} \right) \right). \end{aligned} \quad (18)$$

Note that  $\pi$  only affects to the  $D_{KL}$  term. Then, optimal policy is obtained analytically:

$$\pi^*(\mathbf{x}'|\mathbf{x}) = \frac{p(\mathbf{x}'|\mathbf{x}) z(\mathbf{x}')}{\mathcal{G}[z](\mathbf{x})}. \quad (19)$$

Substituting (19) into (18) yields the linear Bellman equation as:

$$\exp(-hc) z(\mathbf{x}) = \exp(-hq(\mathbf{x})) \mathcal{G}[z](\mathbf{x}). \quad (20)$$

Let  $\mathbf{z}$  be the  $|X|$ -dimensional column vector of associated desirability function. Then the linear Bellman equation (20) can be expressed in a matrix form

$$\lambda \mathbf{z} = Q P \mathbf{z}, \quad (21)$$

where  $\lambda = \exp(-hc)$  is the largest eigenvalue and  $Q = \text{diag}(\exp(-hq(\mathbf{x})))$  [2]. Note that (21) is an eigenvalue problem with  $|X| \times |X|$  matrix. There are various methods to solve an eigenvalue problem; one general way is power iteration method. An interesting point is the power iteration method for this problem is an exact counterpart of the value iteration for standard MDPs but the convergence of power iteration is faster [3]. Despite this advantage, however, the problem is getting intractable as the size of  $X$  increases and the efficient solution method is essential.

##### B. (Global) Planning with Compressed MDPs

Rather than solving original  $|X| \times |X|$  eigenvalue problem, we can treat lower-dimensional coarsened problem. Suppose a set of “abstract-state” at level  $j$ ,  $\Phi_j$ , is utilized as a set of bases for the original problem, which means the problem is viewed in a lower resolution with  $2^{j-1} - 1$  time scale. Then,  $\mathbf{z}$  is approximated as a linear combination of this set:

$$\hat{\mathbf{z}}_j = \Phi_j \mathbf{w}_j, \quad (22)$$

and the original problem (21) is also written in these bases:

$$\hat{\lambda}_j \Phi_j \mathbf{w}_j = Q P \Phi_j \mathbf{w}_j. \quad (23)$$

Since the columns of  $\Phi_j$  are orthogonal, multiplying both sides of (23) by  $\Phi_j'$  yields a compressed problem:

$$\hat{\lambda}_j \mathbf{w}_j = M_j \mathbf{w}_j, \quad (24)$$

where  $M_j := [Q P]_{\Phi_j}^{\Phi_j} = \Phi_j' Q P \Phi_j$  is  $|X_j| \times |X_j|$  matrix with  $M_0 = Q P$ . Note that the compressed problem (24) is much more tractable than the original problem (21) if  $|X_j| \ll |X|$ .

The hierarchical structure of diffusion wavelet tree can be utilized to solve the problem more efficiently. First,

---

**Algorithm 2** Pseudo-code for Global Planning

---

```

// Recursive compression of the operator
1:  $M_0 \leftarrow QP$ 
2: for  $j = 0, \dots, J-1$  do ▷ from fine to coarse
3:    $M_{j+1} \leftarrow [\Phi_{j+1}]'_{\Phi_j} M_j [\Phi_{j+1}]_{\Phi_j}$ 
4: end for
// Recursive solution of MDP
5:  $\mathbf{w}_J \leftarrow \text{POWERITERATION}(M_J, \mathbf{1})$ 
6: for  $j = J-1, \dots, l$  do ▷ from coarse to fine
7:    $\tilde{\mathbf{w}}_j \leftarrow [\Phi_{j+1}]_{\Phi_j} \mathbf{w}_{j+1}$ 
8:    $\mathbf{w}_j \leftarrow \text{POWERITERATION}(M_j, \tilde{\mathbf{w}}_j)$ 
9: end for
10: return  $\hat{\mathbf{z}}_l \leftarrow \Phi_l \mathbf{w}_l$ 

```

---

the matrix  $M_j$  can be computed recursively as  $M_j = [\Phi_{j+1}]'_{\Phi_j} M_{j+1} [\Phi_{j+1}]_{\Phi_j}$ . Also, if an iterative method like a power iteration method is used to solve eigenvalue problem, the solution of  $(j+1)$ th level,  $\mathbf{w}_{j+1}$ , can provide a *warm-start* point to the  $j$ th level problem; that is, the iteration starts from  $\tilde{\mathbf{w}}_j = [\Phi_{j+1}]_{\Phi_j} \mathbf{w}_{j+1}$  by unpacking the solution of next level. If  $\hat{\mathbf{z}}_j$  is not sharply changed through the scale  $j$ , this warm-start will significantly reduce the number of iteration. In summary, in order to solve the global planning at scale  $l$ , the matrix  $M_j$  are computed from finest to coarsest level (from 0 to  $J$ ) and the approximated problem is solved from coarsest to finer level (from  $J$  to  $l$ ), recursively. The pseudo-code of this procedure is shown in Algorithm 2.

### C. (Local) Supplementary/Detailed Planning

In the global planning phase, the solution of the original MDP is approximated using the bases at scale  $l$ . The exact solution can be reconstructed by adding the wavelet functions in lower scales,  $\Psi_{0:l-1}$ , as supplementary bases since  $V_0 = V_l \oplus W_{l-1} \oplus W_{l-2} \oplus \dots \oplus W_0$ . (Note that  $\Phi_l$  and  $\Psi_{0:l-1}$  contain  $|X_l|$  and  $|X| - |X_l|$  bases, respectively.) It is obvious that the larger number of bases are used, the more exact solution we will have, but the problem becomes intractable. An appropriate subset of bases needs to be selected from  $\Psi_{1:l-1}$ . Which bases in  $\Psi_{0:l-1}$  are valuable to improve quality of the solution? The wavelet bases are built as being well-localized. Therefore, the above question can be restated as follow: “which regions of the domain should be treated more extensively?”

In this work, we choose and utilize the supplementary bases which are localized in the region the agent *more likely* visits during  $k_{LP}$ -steps with the approximated optimal policy. And the supplementary bases will be discarded and adaptively re-chosen for every  $k_{LP}$ -steps. Define a weighting of states as:

$$d^{\pi^*}(\mathbf{x}) = \frac{1}{k_{LP}} \sum_{k=1}^{k_{LP}} Pr(\mathbf{x}[k] = \mathbf{x} | \mathbf{x}[0] = \mathbf{x}_{cur}, \pi^*), \quad \forall \mathbf{x} \in X \quad (25)$$

where  $\mathbf{x}_{cur} \in \chi$  is a current state of a robot; of course,  $\mathbf{x}_{cur}$  is not one of discrete points, i.e.  $\mathbf{x} \notin X_{cur}$ .  $Pr(\mathbf{x}[1] = \mathbf{x} | \mathbf{x}[0] = \mathbf{x}_{cur}, \pi^*)$  is obtained as follows. First, the passive

---

**Algorithm 3** Pseudo-code for Local Planning

---

```

1:  $P^* \leftarrow \text{diag}(P\hat{\mathbf{z}}_l)^{-1} P \text{diag}(\hat{\mathbf{z}}_l)$ 
2:  $\mathbf{p} \leftarrow \left[ \frac{\mathcal{N}(\mathbf{x}_m; \mu, \Sigma)}{\sum_{m'} \mathcal{N}(\mathbf{x}_{m'}; \mu, \Sigma)} \right]$  ▷  $\mathbf{p}$ :  $|X|$ -dim row vector
3:  $\bar{\mathbf{p}} \leftarrow \mathbf{p} [\Phi_l]_{\Phi_0}$  ▷  $\bar{\mathbf{p}}$ :  $|X_l|$ -dim row vector
4:  $\bar{P}^* \leftarrow [\Phi_l]_{\Phi_0}' P^* [\Phi_l]_{\Phi_0}$ 
5:  $\bar{\mathbf{d}} \leftarrow \bar{\mathbf{p}} / k_{LP}$ 
6: for  $t = 2, \dots, k_{LP}$  do
7:    $\bar{\mathbf{p}} \leftarrow \bar{\mathbf{p}} \bar{P}^*$ 
8:    $\bar{\mathbf{d}} \leftarrow \bar{\mathbf{d}} + \bar{\mathbf{p}} / k_{LP}$ 
9: end for
10:  $\mathbf{d} \leftarrow \bar{\mathbf{d}} [\Phi_l]_{\Phi_0}'$  ▷  $\mathbf{d}$ :  $|X|$ -dim row vector
11:  $\mathbf{s} \leftarrow \mathbf{d} [|\Psi_{0:(l-1)}|]_{\Phi_0}$  ▷  $\mathbf{s}$ :  $(|X| - |X_l|)$ -dim row vector
12: Choose subset of wavelet bases,  $\Psi_{LP}$ , from  $\Psi_{0:(l-1)}$  based on the score,  $\mathbf{s}$ .
13:  $\mathbf{w} \leftarrow \text{POWERITER} \left( \begin{bmatrix} M_l & \Phi_l' M_0 \Psi_{LP} \\ \Psi_{LP}' M_0 \Phi_l & \Psi_{LP}' M_0 \Psi_{LP} \end{bmatrix}, \begin{bmatrix} \mathbf{w}_l \\ \mathbf{0} \end{bmatrix} \right)$ 
14: return  $\hat{\mathbf{z}} \leftarrow [\Phi_l \quad \Psi_{LP}] \mathbf{w}$ 

```

---

transition probability,  $p(\mathbf{y} | \mathbf{x}_{cur})$ ,  $\forall \mathbf{y} \in X$ , is computed using the Gaussian approximation as in (9). The optimal policy (19) states that the transition probability to  $\mathbf{x}'$  is proportional to the desirability of that state as well as the transition probability without any control. In a similar way, we obtain the (approximate) transition probability as:

$$Pr(\mathbf{x}[1] = \mathbf{x} | \mathbf{x}[0] = \mathbf{x}_{cur}, \pi^*) = \frac{p(\mathbf{x} | \mathbf{x}_{cur}) z(\mathbf{x})}{\sum_{\mathbf{x} \in X} p(\mathbf{x} | \mathbf{x}_{cur}) z(\mathbf{x})}. \quad (26)$$

The probability for other  $k$  can be simply computed using the optimal transition probability matrix  $P^* = \text{diag}(P\mathbf{z})^{-1} P \text{diag}(\mathbf{z})$  as:

$$Pr(\mathbf{x}[k] = \mathbf{x}_m | \mathbf{x}[0] = \mathbf{x}_{cur}, \pi^*) = \sum_{n'} Pr(\mathbf{x}[k-1] = \mathbf{x}_{n'} | \mathbf{x}[0] = \mathbf{x}_{cur}, \pi^*) P_{n'm}^*. \quad (27)$$

Then, all wavelet functions are scored how overlapped with  $d^{\pi^*}(\mathbf{x})$  as:

$$\mathbf{s} = \mathbf{d} [|\Psi_{0:(l-1)}|]_{\Phi_0}, \quad (28)$$

where  $\mathbf{d}$  and  $\mathbf{s}$  are  $|X|$  and  $(|X| - |X_l|)$ -dimensional row vectors each of which components corresponds to the weighting of each state and the score of each basis, respectively. Finally, subset of wavelet bases,  $\Psi_{LP}$ , is selected from  $\Psi_{0:(l-1)}$  based on the score  $\mathbf{s}$  and  $\mathbf{z}$  is re-approximated using the new basis set.

The procedure computing  $d^{\pi^*}(\mathbf{x})$  operated in the original discrete state space requires  $|X|$ -dimensional matrix-vector operations. However, the policy obtained from  $\hat{\mathbf{z}}_l$  is only valid between “abstract-states”,  $\{\phi_{l,k}\}_{k \in X_l}$ . Using this insight, the procedure can be done in the compressed form with  $\Phi_l$ : we compress the operator as  $\bar{P}^* = [\Phi_l]_{\Phi_0}' P^* [\Phi_l]_{\Phi_0}$ , compute  $d^{\pi^*}(\mathbf{x})$  in the compressed domain and unpack the result (as shown in line 3 - 10 in Algorithm 3). As a result,  $|X|$ -dimensional operations are reduced to  $|X_l|$ -dimensional operations.

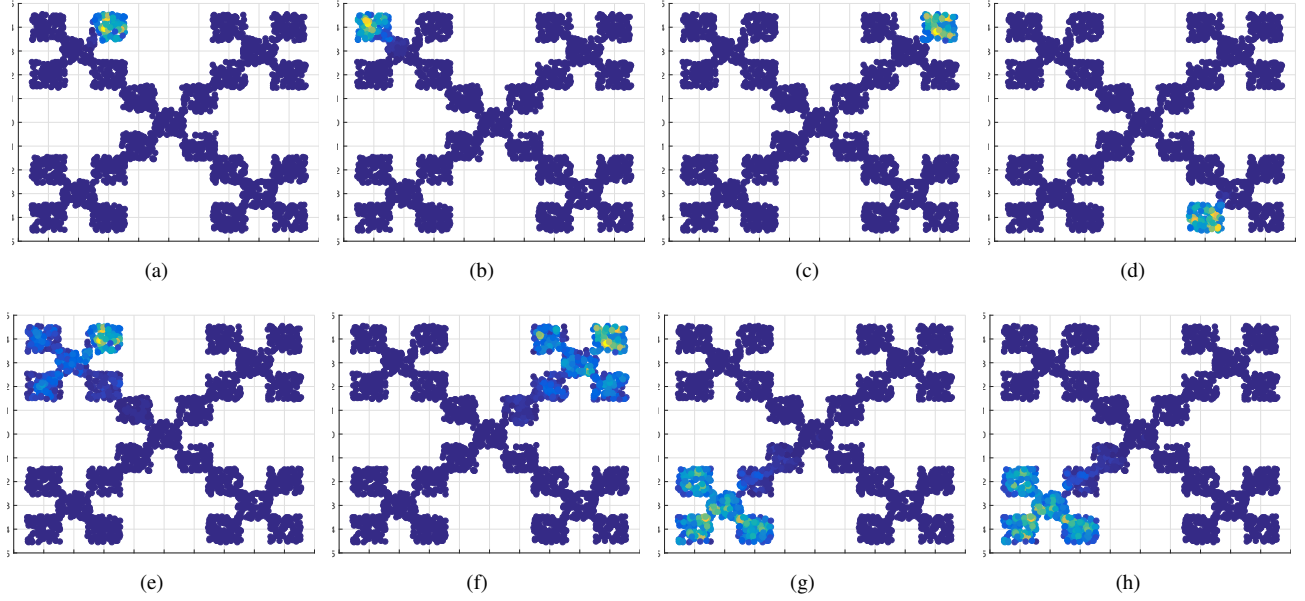


Fig. 2. Some scaling functions at (a)-(d) level 8 and (e)-(h) level 12 in fractal environment example.

## V. RECEDING HORIZON CONTROL IN CONTINUOUS TIME

Using the desirability function on the set of sample states, continuous-time optimal control sequence  $\mathbf{u}^*(t)$  needs to be computed. We will compute and apply the continuous optimal control sequence for the interval  $\tau = h \times k_{RHC}$  in a receding horizon control fashion.

The control can be computed by matching the 1st order moment of original SDE (1) and the optimal policy (19) for the MDP. First, the optimal transition probability for  $k_{RHC}$ -steps,  $Pr(\mathbf{x}[k_{RHC}] = \mathbf{y} | \mathbf{x}[0] = \mathbf{x}_{cur}, \pi^*)$ , is computed in the same way as (26)-(27). If the robot follows the optimal policy (19), the expected state after  $\tau$  will be

$$\mathbf{y}_{new} = \sum_{\mathbf{y} \in X} \mathbf{y} Pr(\mathbf{x}[k_{RHC}] = \mathbf{y} | \mathbf{x}[0] = \mathbf{x}_{cur}, \pi^*). \quad (29)$$

Then, the control that matches the state mean of the robot after  $h$  with  $\mathbf{y}_{new}$  is computed as following proposition.

**Proposition 1:** Consider the linearized 1st order moment dynamics of (1),

$$\dot{\mu}_c(t) = A\mu_c(t) + G\mathbf{u}(t) + \mathbf{c}, \quad (30)$$

where  $A = \frac{d\mathbf{f}}{d\mathbf{x}}|_{\mathbf{x}=\mathbf{x}_{cur}}$ ,  $G = G(\mathbf{x}_{cur})$  and  $\mathbf{c} = \mathbf{f}(\mathbf{x}_{cur}) - A\mathbf{x}_{cur}$ . Suppose that the initial state is given by  $\mu_c(0) = \mathbf{x}_{cur}$  and the control sequence,

$$\mathbf{u}^*(t) = -\sigma^2 G' e^{A'(\tau-t)} \Sigma(\tau)^{-1} (\mu(\tau) - \mathbf{y}_{new}), \quad (31)$$

is applied to (30) for  $t \in [0, \tau]$ , where  $\mu(\tau)$  and  $\Sigma(\tau)$  are solutions of (10) and (11) with  $\mu(0) = \mathbf{x}_{cur}$ ,  $\Sigma(0) = 0$ , respectively. Then,  $\mu_c(\tau) = \mathbf{y}_{new}$ .

*Proof:* The solution of (30) is given by

$$\mu_c(\tau) = e^{A\tau} \mu_c(0) + \int_0^\tau e^{A(\tau-t)} (G\mathbf{u} + \mathbf{c}) dt. \quad (32)$$

Substituting (31) into (32) yields

$$\begin{aligned} \mu_c(\tau) &= e^{A\tau} \mu_c(0) + \int_0^\tau e^{A(\tau-t)} \mathbf{c} dt \\ &\quad - \sigma^2 \int_0^\tau e^{A(\tau-t)} G G' e^{A'(\tau-t)} dt \Sigma(\tau)^{-1} (\mu(\tau) - \mathbf{y}_{new}) \\ &= \mu(\tau) - \Sigma(\tau) \Sigma(\tau)^{-1} (\mu(\tau) - \mathbf{y}_{new}) \\ &= \mathbf{y}_{new} \end{aligned} \quad (33)$$

We utilize the analytic solutions of (10) and (11):

$$\begin{aligned} \mu(\tau) &= e^{A\tau} \mu(0) + \int_0^\tau e^{A(\tau-t)} \mathbf{c} dt, \\ \Sigma(\tau) &= \int_0^\tau e^{A(\tau-t)} B B' e^{A'(\tau-t)} dt, \end{aligned} \quad (34)$$

and  $B B' = \sigma^2 G G'$ . ■

**Remark 2:** It is also worth noting that  $\mathbf{u}^*(t)$  is the solution of a deterministic optimal control problem of dynamic system (30) with cost function

$$J = \int_0^\tau q(\mathbf{x}_{cur}) + \frac{1}{2\sigma^2} \mathbf{u}(t)' \mathbf{u}(t) dt,$$

subject to  $\mu_c(0) = \mathbf{x}_{cur}$  and  $\mu_c(\tau) = \mathbf{y}_{new}$ , which is also called an affine-quadratic optimal control problem.

Finally, we'd like to emphasize that the time interval  $\tau$  should be determined to be small as the linearization effect on the mean dynamics (30) is not too significant.

## VI. NUMERICAL EXAMPLES

### A. Robot Navigation in Fractal Shape Environment

For the first example, we consider a simple two-dimensional stochastic single integrator in the fractal-like environment. The environment consists of 5 group of rooms where one group is made of 5 rooms as shown in Fig. 2; one can observe that the environment has 2 level self-similarity,



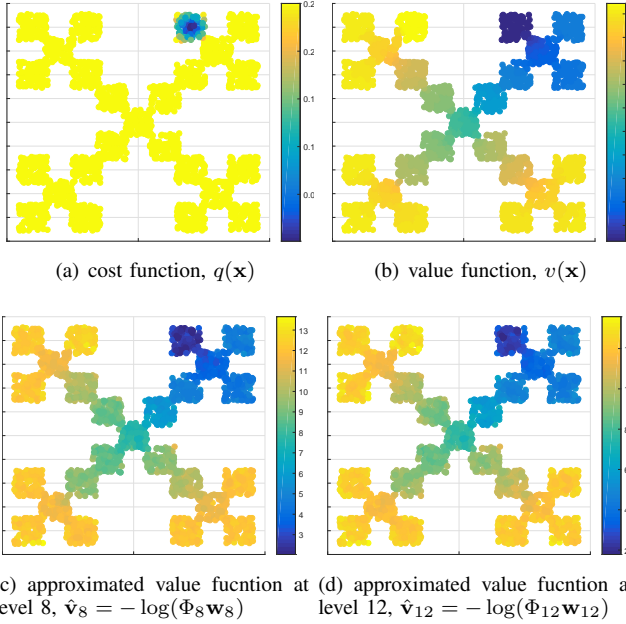


Fig. 3. Fractal environment example. Cost, value and approximated value function at level 8 and 12.

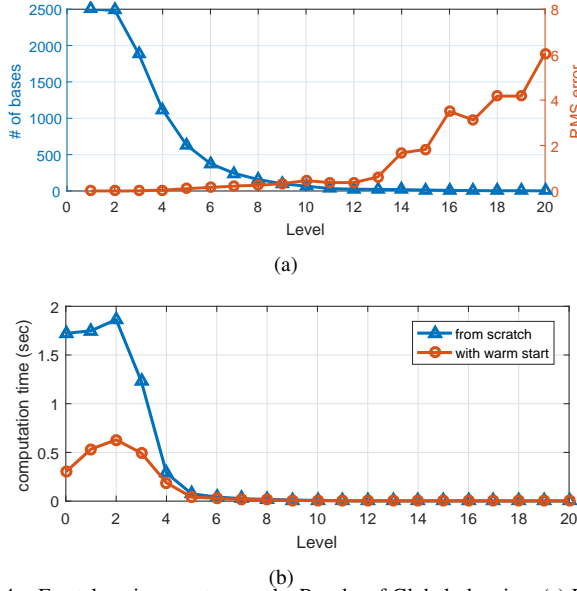


Fig. 4. Fractal environment example. Results of Global planning. (a) RMS errors are measured between  $\mathbf{v}$  and  $\hat{\mathbf{v}}$ . (b) Eigenvalue problems are solved using MATLAB build-in function, *eigs*.

which makes the problem have a multiscale nature. The dynamics is given by:

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}, \quad G(\mathbf{x}) = I_2$$

i.e. the position of a robot in the configuration space,  $\mathbf{x} \in \chi$ , is controlled by the velocity input,  $\mathbf{u} \in \mathbb{R}^2$ . We set  $h = 0.01$  and  $\sigma = 1$ . In this setting,  $\mu(h) = 0$  and  $\Sigma(h) = hI_2$  are obtained analytically, then the Markov chain is constructed by simple Gaussian probability with Euclidean distance, which has been extensively studied in the graph Laplacian-based solution approach for MDPs [11]–[13]. Our stochastic optimal control problem formulation can be viewed as a generalization of them, since it can treat more

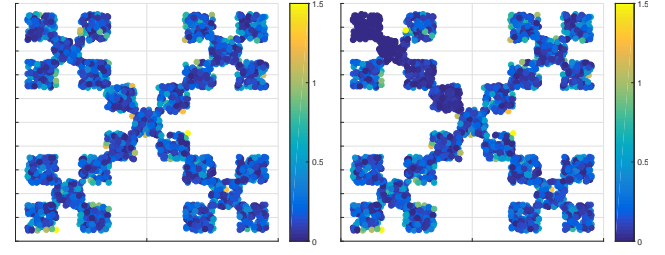


Fig. 5. Fractal environment example. Results of local planning. Errors of value function is measured as  $|v(\mathbf{x}) - \hat{v}(\mathbf{x})|$ . The robot is assumed to locate in the center of the most left and upper room.

general dynamics and cost function.

In order to discretize the state space, 100 samples are obtained from each room, therefore there are 2500 discrete state in total. Fig. 2 shows the abstraction results; some scaling functions in the diffusion wavelet tree at level 8 and 12. It is observed that at level 8 and 12, scaling functions roughly represent each small room and one group of 5 rooms, respectively. The cost function of the problem is depicted in Fig. 3(a); it denotes the goal of the robot in this environment. The optimal value function on original bases and its approximations on the coarse bases are shown in Fig. 3(b)-(d). It is observed from Fig. 4(a) that as the scale level increases, the number of bases are significantly reduced but the value function is quite well-approximated. We solve the eigenvalue problems (24) using MATLAB built-in function, *eigs*, with/without the warm start. Fig. 4(b) shows that the approximate solution of coarser level greatly helps rapid convergence of eigenvalue problem solver. Finally, the result of local planning is shown in Fig. 5; the robot is assumed to locate in the center of the most left and upper room. After the global planning, the approximated value function contains some error (see Fig. 5(a)) but coarse level of plan can be reconstructed (see Fig. 3(d)). Then it can evaluates which regions will be visited and make a detailed plan for those regions; Fig. 3(b) shows that value function error in the room located on the way to goal is significantly reduced after the local planning.

### B. Inverted Pendulum Metronome

The second example deals with pendulum metronome task which has been dealt in [3]. This example has the optimal policy which drives the system to the limit cycle attractor. The states and the control input of the system are pendulum's orientation,  $\theta$ , angular velocity,  $\omega$ , and torque,  $T$ . Then, the (inverted) pendulum dynamics is given by:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \omega \\ \sin(\theta) \end{bmatrix}, \quad G(\mathbf{x}) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

We set  $h = 0.1$ ,  $\sigma = 3$  and sample 3000 discrete states to construct the Markov chain. In addition, the collision

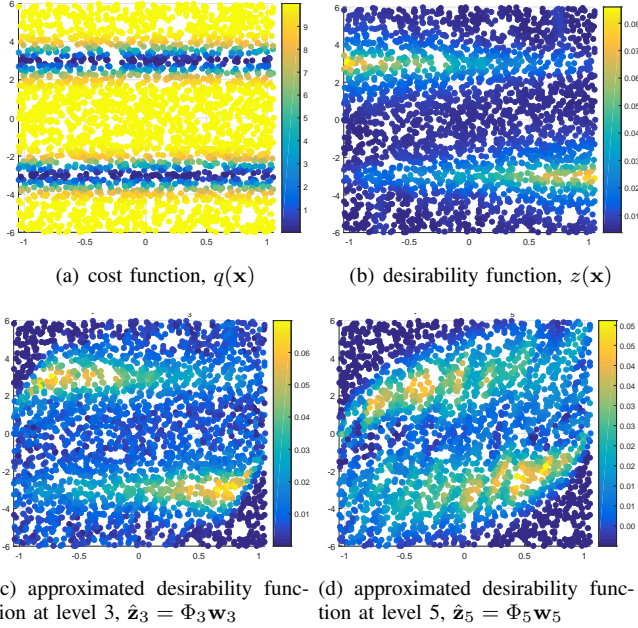


Fig. 6. Pendulum metronome example. In order to represent the (approximate) desirability functions, (b) 3000 (c) 989 (d) 52 basis functions are used.  $x$  and  $y$  axes represent  $\theta$  and  $\omega$ , respectively.

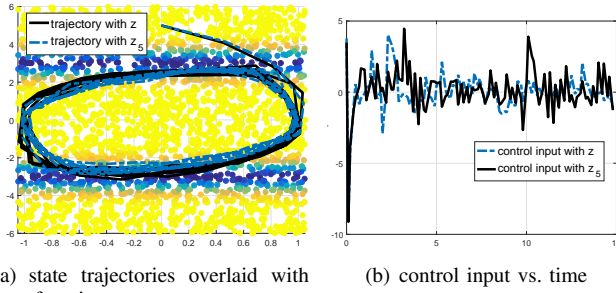


Fig. 7. Pendulum metronome example. Resulting continuous state and control input trajectories. To see resulting policy and limit cycle clearly, we simulate the trajectories without disturbance.

dynamics is modeled as in [3],

$$\begin{aligned} \text{if } \theta[k] < -\frac{\pi}{3}, \text{ then } \theta[k+1] &= -\frac{\pi}{3}, \omega[k+1] = 0, \\ \text{if } \theta[k] > \frac{\pi}{3}, \text{ then } \theta[k+1] &= \frac{\pi}{3}, \omega[k+1] = 0; \end{aligned}$$

that is, shock absorbers cause inelastic collisions on both sides of the orientation limit.

Fig. 6 shows the cost function and desirability functions on original and coarse bases; the objective of the motion is to maintain the desired speed, 3 rad/sec. It is seen that the desirability after collision is biggest since the pendulum need to accelerate in order to achieve the desired speed; once the speed is achieved, its passive dynamics lets the pendulum maintain its speed until next collision occurs. Note that such kinds of policy are observed from the approximated desirability at higher scales; especially, only 52 bases can construct this tendency at level 5. Continuous-time control is computed and applied in a receding horizon fashion, and the resulting trajectories are depicted in Fig. 7. It is observed that the desirability functions on 3000 (delta) bases set and on 52 level-5 bases set result almost same limit cycle attractors.

## VII. CONCLUSIONS

This work has proposed a multiscale framework to solve a class of continuous-time, continuous-space stochastic optimal control problems in a complex environment. Using bases obtained by diffusion wavelet method, the problem has been solved efficiently: the global plan is computed with coarse resolution and the detailed plan is obtained for only important regions. In addition, combined with a receding-horizon scheme in execution of the optimal control solution, the proposed method can generate continuous control sequence for robot motion. Numerical examples have demonstrated that the optimal solution can be expressed in very low-dimensional parametric space since the multi-scale abstraction provides meaningful basis functions.

Note that once a Markov chain is constructed, a complexity of domain geometry and size of decision space do not affect the computational complexity of the proposed framework. Also, because a Markov chain only needs to contain the information about dynamics and domain, same abstraction can be re-utilized to various tasks. We expect that this will allow us treating challenging planning & control problems where extensive robotic research is being investigated.

## ACKNOWLEDGMENT

This work was supported by Agency for Defense Development (under in part contract #UD140053JD and in part contract #UD150047JD).

## REFERENCES

- [1] H. J. Kappen, "Path integrals and symmetry breaking for optimal control theory," *Journal of statistical mechanics: theory and experiment*, vol. 2005, no. 11, p. P11011, 2005.
- [2] E. Todorov, "Efficient computation of optimal actions," *Proceedings of the national academy of sciences*, 2009.
- [3] —, "Eigenfunction approximation methods for linearly-solvable optimal control problems," in *2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*. IEEE, 2009, pp. 161–168.
- [4] J.-S. Ha and H.-L. Choi, "A topology-guided path integral approach for stochastic optimal control," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.
- [5] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1433–1440.
- [6] J. C. Goswami and A. K. Chan, *Fundamentals of wavelets: theory, algorithms, and applications*. John Wiley & Sons, 2011, vol. 233.
- [7] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 53–94, 2006.
- [8] M. Maggioni and S. Mahadevan, "Fast direct policy evaluation using multiscale analysis of markov diffusion processes," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006.
- [9] S. Mahadevan and M. Maggioni, "Value function approximation with diffusion wavelets and laplacian eigenfunctions," in *Advances in neural information processing systems*, 2005, pp. 843–850.
- [10] H. Kushner and P. G. Dupuis, *Numerical methods for stochastic control problems in continuous time*. Springer Science & Business Media, 2013, vol. 24.
- [11] S. Mahadevan and M. Maggioni, "Proto-value functions: A laplacian framework for learning representation and control in markov decision processes," *Journal of Machine Learning Research*, vol. 8, no. Oct, pp. 2169–2231, 2007.
- [12] D. S. Corneil and W. Gerstner, "Attractor network dynamics enable preplay and rapid path planning in maze-like environments," in *Advances in Neural Information Processing Systems*, 2015.
- [13] Y. F. Chen, S.-Y. Liu, M. Liu, J. Miller, and J. P. How, "Motion planning with diffusion maps," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.